

Application for United States Letters Patent

for

MEMORY MANAGEMENT SYSTEM AND METHOD

PROVIDING INCREASED MEMORY ACCESS SECURITY

by

Geoffrey S. Strongin

Brian C. Barnes

Rodney W. Schmidt

EXPRESS MAIL MAILING LABEL

NUMBER EL504116674US

DATE OF DEPOSIT December 5, 2001

I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington D.C. 20231.



Signature

MEMORY MANAGEMENT SYSTEM AND METHOD

PROVIDING INCREASED MEMORY ACCESS SECURITY

BACKGROUND OF THE INVENTION

5 1. REFERENCES

This patent application is related to a co-pending patent application serial no.

_____ (Attorney Reference Number 2000.055400/TT4074) entitled "System and Method for Handling Device Accesses to a Memory Providing Increased Memory Access Security" by Geoffrey S. Strongin, Brian C. Barnes, and Rodney W. Schmidt, filed on the same day as the present patent application.

2. FIELD OF THE INVENTION

This invention relates generally to memory management systems and methods, and, more particularly, to memory management systems and methods that provide protection for data stored within a memory.

3. DESCRIPTION OF THE RELATED ART

A typical computer system includes a memory hierarchy to obtain a relatively high level of performance at relatively low cost. Instructions of several different software programs are typically stored on a relatively large but slow non-volatile storage unit (e.g., a disk drive unit). When a user selects one of the programs for execution, the instructions of the selected program are copied into a main memory unit (e.g., random access memory (RAM)), and a central processing unit (CPU) obtains the instructions of the selected program from the main memory unit. A well-known virtual memory management technique allows the CPU to access data structures larger in size than that of the main memory unit by storing only a portion of the data structures within the main memory unit at any given time.

Remainders of the data structures are stored within the relatively large but slow non-volatile storage unit, and are copied into the main memory unit only when needed.

Virtual memory is typically implemented by dividing an address space of the CPU

5 into multiple blocks called page frames or “pages.” Only data corresponding to a portion of the pages is stored within the main memory unit at any given time. When the CPU generates an address within a given page, and a copy of that page is not located within the main memory unit, the required page of data is copied from the relatively large but slow non-volatile storage unit into the main memory unit. In the process, another page of data may be
10 copied from the main memory unit to the non-volatile storage unit to make room for the required page.

15

20

25

The popular 80x86 (x86) processor architecture includes specialized hardware elements to support a protected virtual address mode (i.e., a protected mode). Figs. 1-3 will now be used to describe how an x86 processor implements both virtual memory and memory protection features. Fig. 1 is a diagram of a well-known linear-to-physical address translation mechanism 100 of the x86 processor architecture. An address translation mechanism 100 is embodied within an x86 processor, and involves a linear address 102 produced within the x86 processor, a page table directory (i.e., a page directory) 104, multiple page tables including a page table 106, multiple page frames including a page frame 108, and a control register (CR3) 110. The page directory 104 and the multiple page tables are paged memory data structures created and maintained by operating system software (i.e., an operating system). The page directory 104 is always located within the memory (e.g., the main memory unit). For simplicity, the page table 106 and the page frame 108 will also be assumed to reside in the memory.

As indicated in Fig. 1, the linear address 102 is divided into three portions to accomplish the linear-to-physical address translation. The highest ordered bits of the CR3 110 are used to store a page directory base register. The page directory base register is a base address of a memory page containing the page directory 104. The page directory 104 includes multiple page directory entries, including a page directory entry 112. An upper “directory index” portion of the linear address 102, including the highest ordered or most significant bits of the linear address 102, is used as an index into the page directory 104. The page directory entry 112 is selected from within the page directory 104 using the page directory base address of the CR3 110 and the upper “directory index” portion of the linear address 102.

Fig. 2 is a diagram of a page directory entry format 200 of the x86 processor architecture. As indicated in Fig. 2, the highest ordered (i.e., most significant) bits of a given page directory entry contain a page table base address, where the page table base address is a base address of a memory page containing a corresponding page table. The page table base address of the page directory entry 112 is used to select the corresponding page table 106.

Referring back to Fig. 1, the page table 106 includes multiple page table entries, including a page table entry 114. A middle “table index” portion of the linear address 102 is used as an index into the page table 106, thereby selecting the page table entry 114. Fig. 3 is a diagram of a page table entry format 300 of the x86 processor architecture. As indicated in Fig. 3, the highest ordered (i.e., most significant) bits of a given page table entry contain a page frame base address, where the page frame base address is a base address of a corresponding page frame.

Referring again to Fig. 1, the page frame base address of the page table entry 114 is used to select the corresponding page frame 108. The page frame 108 includes multiple memory locations. A lower or “offset” portion of the linear address 102 is used as an index 5 into the page frame 108. When combined, the page frame base address of the page table entry 114 and the offset portion of the linear address 102 produce the physical address corresponding to the linear address 102, and indicate a memory location 116 within the page frame 108. The memory location 116 has the physical address resulting from the linear-to-physical address translation.

10 Regarding the memory protection features, the page directory entry format 200 of Fig.

2 and the page table entry format 300 of Fig. 3 include a user/supervisor (U/S) bit and a read/write (R/W) bit. The contents of the U/S and R/W bits are used by the operating system to protect corresponding page frames (i.e., memory pages) from unauthorized access. U/S=0 15 is used to denote operating system memory pages, and corresponds to a “supervisor” level of the operating system. The supervisor level of the operating system corresponds to a current privilege level 0 (CPL0) of software programs and routines executed by the x86 processor. U/S>0 (e.g., U/S=1, 2, or 3) is used to indicate user memory pages, and corresponds to a “user” level of the operating system.

20 The R/W bit is used to indicate types of accesses allowed to the corresponding memory page. R/W=0 indicates the only read accesses are allowed to the corresponding memory page (i.e., the corresponding memory page is “read-only”). R/W=1 indicates that both read and write accesses are allowed to the corresponding memory page (i.e., the 25 corresponding memory page is “read-write”).

During the linear-to-physical address translation operation of Fig. 1, the contents of the U/S bits of the page directory entry 112 and the page table entry 114, corresponding to the page frame 108, are logically ANDed to determine if the access to the page frame 108 is authorized. Similarly, the contents of the R/W bits of the page directory entry 112 and the page table entry 114 are logically ANDed to determine if the access to the page frame 108 is authorized. If the logical combinations of the U/S and R/W bits indicate the access to the page frame 108 is authorized, the memory location 116 is accessed using the physical address. On the other hand, if the logical combinations of the U/S and R/W bits indicate that the access to the page frame 108 is not authorized, the memory location 116 is not accessed, and a protection fault indication is signaled.

10
15

20

Unfortunately, the above described memory protection mechanisms of the x86 processor architecture are not sufficient to protect data stored in the memory. For example, any software program or routine executing at the supervisor level (e.g., having a CPL of 0) can access any portion of the memory, and can modify (i.e., write to) any portion of the memory that is not marked "read-only" ($R/W=0$). In addition, by virtue of executing at the supervisor level, the software program or routine can change the attributes (i.e., the U/S and R/W bits) of any portion of the memory. The software program or routine can thus change any portion of the memory marked "read-only" to "read-write" ($R/W=1$), and then proceed to modify that portion of the memory.

25

The protection mechanisms of the x86 processor architecture are also inadequate to prevent errant or malicious accesses to the memory by hardware devices operably coupled to the memory. It is true that portions of the memory marked "read-only" cannot be modified

by write accesses initiated by hardware devices (without the attributes of those portions of the memory first being changed as described above). It is also true that software programs or routines (e.g., device drivers) handling data transfers between hardware devices and the memory typically execute at the user level (e.g., CPL3), and are not permitted access to

5 portions of the memory marked as supervisor level (U/S=0). However, the protection mechanisms of the x86 processor architecture cover only device accesses to the memory performed as a result of instruction execution (i.e., programmed input/output). A device driver can program a hardware device having bus mastering or DMA capability to transfer data from the device into any portion of the memory accessible by the hardware device. For

10 example, it is relatively straightforward to program a floppy disk controller to transfer data from a floppy disk directly into a portion of the memory used to store the operating system.

SUMMARY OF THE INVENTION

A memory management unit (MMU) is disclosed for managing a memory storing data arranged within a multiple memory pages. The memory management unit includes a security check unit receiving a physical address within a selected memory page, and security attributes of the selected memory page. The security check unit uses the physical address to access one or more security attribute data structures located in the memory to obtain an additional security attribute of the selected memory page. The security check unit generates a fault signal dependent upon the security attributes of selected memory page and the

20 additional security attribute of the selected memory page.

The security attributes of the selected memory page may include, for example, a user/supervisor (U/S) bit and a read/write (R/W) bit as defined by the x86 processor architecture. In this situation, U/S=0 indicates the selected memory page is an operating

25 system memory page and corresponds to a supervisor level of the operating system, and

U/S=1 indicates the selected memory page is a user memory page and corresponds to a user level of the operating system. R/W=0 indicates only read accesses are allowed to the selected memory page, and R/W=1 indicates that both read and write accesses are allowed to the selected memory page.

5

The one or more security attribute data structures may include a security attribute table directory and one or more security attribute tables. The security attribute table directory may include multiple entries, and each entry of the security attribute table directory may include a present bit and a security attribute table base address field. The present bit may indicate whether or not a security attribute table corresponding to the security attribute table directory entry is present in the memory. The security attribute table base address field may be reserved for a base address of the security attribute table corresponding to the security attribute table directory entry.

10
15

The one or more security attribute tables may include multiple entries. Each entry of the security attribute table may include, for example, a secure page (SP) bit indicating whether or not a corresponding memory page is a secure page. The additional security attribute of the selected memory page may include a secure page (SP) bit indicating whether or not the selected memory page is a secure page.

20

25

The linear address may be produced during execution of an instruction residing within a first memory page. The security check unit may be coupled to receive a current privilege level (CPL) of a task including the instruction. The security check logic may obtain an additional security attribute of the first memory page from the one or more security attribute data structures. The security check logic may generate the fault signal dependent upon the

CPL of the task including the instruction, the additional security attribute of the first memory page, the security attributes of the selected memory page, and the additional security attribute of the selected memory page. The additional security attribute of the first memory page may include a secure page (SP) bit indicating whether or not the first memory page is a secure page. The fault signal may be a page fault signal as defined by the x86 processor architecture.

A central processing unit (CPU) is described including an execution unit and the above described memory management unit (MMU). The execution unit is coupled to a memory, and fetches instructions from the memory and executes the instructions. A computer system is disclosed including a memory for storing data including instructions, a central processing unit (CPU) including an execution unit coupled to the memory, and the above described memory management unit (MMU). The execution unit fetches instructions from the memory and executes the instructions.

10
11
12
13
14
15
16
17
18
19
20

A memory management unit is disclosed for managing a memory storing data arranged within a multiple memory pages. The memory management unit includes a paging unit coupled to the memory and to receive a linear address. The memory management unit is configured to use the linear address to produce a physical address within a selected memory page. The paging unit uses the linear address to access one or more paged memory data structures located in the memory to obtain security attributes of the selected memory page. The paging unit produces a fault signal dependent upon the security attributes of the selected memory page.

The paging unit includes a security check unit coupled to receive the physical address and the security attributes of the selected memory page. The security check unit uses the physical address of the selected memory page to access one or more security attribute data structures located in the memory to obtain an additional security attribute of the selected 5 memory page. The security check unit generates the fault signal dependent upon the security attributes of selected memory page and the additional security attribute of the selected memory page.

10 The paging unit may produce the physical address of the selected memory page during execution of an instruction residing within a first memory page. The physical address within the selected memory page may include a base address and an offset. The paging unit may obtain the base address from the one or more paged memory data structures. The one or more paged memory data structures may include, for example, a page directory and one or more page tables as defined by the x86 processor architecture. The security attributes of the selected memory page may include a user/supervisor (U/S) bit and a read/write (R/W) bit as defined by the x86 processor architecture.

15 The paging unit may receive a security attribute of the instruction, and may produce the fault signal dependent upon the security attribute of the instruction and the security attributes of the selected memory page. The security attribute of the instruction may include 20 a current privilege level (CPL) of a task including the instruction as defined by the x86 processor architecture.

25 The one or more security attribute data structures may include a security attribute table directory and one or more security attribute tables. The security attribute table directory

may include multiple entries, and each entry of the security attribute table directory may include a present bit and a security attribute table base address field. The present bit may indicate whether or not a security attribute table corresponding to the security attribute table directory entry is present in the memory. The security attribute table base address field may

5 be reserved for a base address of the security attribute table corresponding to the security attribute table directory entry.

The one or more security attribute tables may include multiple entries, and each entry of the security attribute table may include a secure page (SP) bit indicating whether or not a

10 corresponding memory page is a secure page. The additional security attribute of the selected memory page may include a secure page (SP) bit indicating whether or not the selected memory page is a secure page.

The security check unit may receive the CPL of the task including the instruction.

15 The security check logic may obtain an additional security attribute of the first memory page including the instruction from the one or more security attribute data structures. The security check unit may generate the fault signal dependent upon the CPL of the task including the instruction, the additional security attribute of the first memory page including the instruction, the security attributes of the selected memory page, and the additional security

20 attribute of the selected memory page. The additional security attribute of the first memory page may include a secure page (SP) bit indicating whether or not the first memory page is a secure page. The fault signal may be a page fault signal as defined by the x86 processor architecture.

A method is described for providing access security for a memory used to store data arranged within multiple memory pages. The method includes receiving a linear address produced during execution of an instruction and a security attribute of the instruction, wherein the instruction resides in a first memory page. The linear address is used to access 5 one or more paged memory data structures located in the memory to obtain a base address of a selected memory page and security attributes of the selected memory page. If the security attribute of the instruction and the security attributes of the selected memory page indicate the access is authorized, the base address of the selected memory page is combined with an offset to produce a physical address within the selected memory page. A fault signal is generated if 10 the security attribute of the instruction and the security attributes of the selected memory page indicate the access is not authorized.

10
11
12
13
14
15

One or more security attribute data structures, located in the memory, are accessed using the physical address of the selected memory page to obtain an additional security attribute of the first memory page and an additional security attribute of the selected memory page. The fault signal is generated dependent upon the security attribute of the instruction, the additional security attribute of the first memory page, the security attributes of the selected memory page, and the additional security attribute of the selected memory page.

20 The one or more paged memory data structures may include a page directory and one or more page tables as defined by the x86 processor architecture. The security attribute of the instruction may include a current privilege level (CPL) of a task including the instruction as defined by the x86 processor architecture. The security attributes of the selected memory page may include a user/supervisor (U/S) bit a read/write (R/W) bit as defined by the x86

processor architecture. The fault signal may be a page fault signal as defined by the x86 processor architecture.

The additional security attribute of the first memory page may include a secure page

- 5 (SP) bit indicating whether or not the first memory page is a secure page. The additional security attribute of the selected memory page may also include a secure page (SP) bit indicating whether or not the selected memory page is a secure page. The one or more security attribute data structures may include a security attribute table directory and one or more security attribute tables.

10 BRIEF DESCRIPTION OF THE DRAWINGS

The invention may be understood by reference to the following description taken in conjunction with the accompanying drawings, in which like reference numerals identify similar elements, and in which:

15 Fig. 1 is a diagram of a well-known linear-to-physical address translation mechanism of the x86 processor architecture;

Fig. 2 is a diagram of a page directory entry format of the x86 processor architecture;

20 Fig. 3 is a diagram of a page table entry format of the x86 processor architecture;

Fig. 4 is a diagram of one embodiment of a computer system including a CPU and a system or “host” bridge, wherein the CPU includes a CPU security check unit (SCU), and wherein the host bridge includes a host bridge SCU;

Fig. 5 is a diagram illustrating relationships between various hardware and software components of the computer system of Fig. 4;

5 Fig. 6 is a diagram of one embodiment of the CPU of the computer system of Fig. 4, wherein the CPU includes a memory management unit (MMU);

Fig. 7 is a diagram of one embodiment of the MMU of Fig. 6, wherein the MMU includes a paging unit, and wherein the paging unit includes the CPU SCU;

10 Fig. 8 is a diagram of one embodiment of the CPU SCU of Fig. 7;

Fig. 9 is a diagram of one embodiment of a mechanism for accessing a security attribute table (SAT) entry of a selected memory page to obtain additional security information of the selected memory page;

Fig. 10 is a diagram of one embodiment of a SAT default register;

Fig. 11 is a diagram of one embodiment of a SAT directory entry format;

20 Fig. 12 is a diagram of one embodiment of a SAT entry format;

Fig. 13 is a diagram of one embodiment of the host bridge of Fig. 4, wherein the host bridge includes the host bridge SCU;

25 Fig. 14 is a diagram of one embodiment of the host bridge SCU of Fig. 13;

Fig. 15 is a flow chart of one embodiment of a first method for managing a memory used to store data arranged within multiple memory pages; and

5 Fig. 16 is a flow chart of one embodiment of a second method for providing access security for a memory used to store data arranged within multiple memory pages.

10 While the invention is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the drawings and are herein described in detail. It should be understood, however, that the description herein of specific embodiments is not intended to limit the invention to the particular forms disclosed, but on the contrary, the intention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

15 Illustrative embodiments of the invention are described below. In the interest of clarity, not all features of an actual implementation are described in this specification. It will, of course, be appreciated that in the development of any such actual embodiment, numerous implementation-specific decisions must be made to achieve the developers' specific goals, such as compliance with system-related and business-related constraints, which will vary 20 from one implementation to another. Moreover, it will be appreciated that such a development effort might be complex and time-consuming, but would nevertheless be a routine undertaking for those of ordinary skill in the art having the benefit of this disclosure.

25 Fig. 4 is a diagram of one embodiment of a computer system 400 including a CPU 402, a system or "host" bridge 404, a memory 406, a first device bus 408 (e.g., a peripheral

component interconnect or PCI bus), a device bus bridge 410, a second device bus 412 (e.g., an industry standard architecture or ISA bus), and four device hardware units 414A-414D. Host bridge 404 is coupled to CPU 402, memory 406, and device bus 408. Host bridge 404 translates signals between CPU 402 and device bus 408, and operably couples memory 406 to CPU 402 and to device bus 408. Device bus bridge 410 is coupled between device bus 408 and device bus 412, and translates signals between device bus 408 and device bus 412. In the embodiment of Fig. 4, device hardware units 414A and 414B are coupled to device bus 408, and device hardware units 414C and 414D are coupled to device bus 412. One or more of the device hardware units 414A-414D may be, for example, storage devices (e.g., hard disk drives, floppy drives, and CD-ROM drives), communication devices (e.g., modems and network adapters), or input/output devices (e.g., video devices, audio devices, and printers).

In the embodiment of Fig. 4, CPU 402 includes a CPU security check unit (SCU) 416, and host bridge 404 includes a host bridge SCU 418. As will be described in detail below, CPU SCU 416 protects memory 406 from unauthorized accesses generated by CPU 402 (i.e., “software-initiated accesses”), and host bridge SCU 418 protects memory 406 from unauthorized accesses initiated by device hardware units 414A-414D (i.e., “hardware-initiated accesses”). It is noted that in other embodiments, host bridge 404 may be part of CPU 402 as indicated in Fig. 4.

20

Fig. 5 is a diagram illustrating relationships between various hardware and software components of computer system 400 of Fig. 4. In the embodiment of Fig. 5, multiple application programs 500, an operating system 502, a security kernel 504, and device drivers 506A-506D are stored in memory 406. Application programs 500, operating system 502, security kernel 504, and device drivers 506A-506D include instructions executed by CPU

402. Operating system 502 provides a user interface and software “platform” on top of which application programs 500 run. Operating system 502 may also provide, for example, basic support functions including file system management, process management, and input/output (I/O) control.

5

Operating system 502 may also provide basic security functions. For example, CPU 402 (Fig. 4) may be an x86 processor which executes instructions of the x86 instruction set. In this situation, CPU 402 may include specialized hardware elements to provide both virtual memory and memory protection features in the protected mode as described above. 10 Operating system 502 may be, for example, one of the Windows® family of operating systems (Microsoft Corp., Redmond, WA) which operates CPU 402 in the protected mode, and uses the specialized hardware elements of CPU 402 to provide both virtual memory and memory protection in the protected mode.

406
408
410
412
414
416
418
420
422
424
426
428
430
432
434
436
438
440
442
444
446
448
450
452
454
456
458
460
462
464
466
468
470
472
474
476
478
480
482
484
486
488
490
492
494
496
498
500
502
504
506
508
510
512
514
516
518
520
522
524
526
528
530
532
534
536
538
540
542
544
546
548
550
552
554
556
558
560
562
564
566
568
570
572
574
576
578
580
582
584
586
588
590
592
594
596
598
600
602
604
606
608
610
612
614
616
618
620
622
624
626
628
630
632
634
636
638
640
642
644
646
648
650
652
654
656
658
660
662
664
666
668
670
672
674
676
678
680
682
684
686
688
690
692
694
696
698
700
702
704
706
708
710
712
714
716
718
720
722
724
726
728
730
732
734
736
738
740
742
744
746
748
750
752
754
756
758
760
762
764
766
768
770
772
774
776
778
780
782
784
786
788
790
792
794
796
798
800
802
804
806
808
810
812
814
816
818
820
822
824
826
828
830
832
834
836
838
840
842
844
846
848
850
852
854
856
858
860
862
864
866
868
870
872
874
876
878
880
882
884
886
888
890
892
894
896
898
900
902
904
906
908
910
912
914
916
918
920
922
924
926
928
930
932
934
936
938
940
942
944
946
948
950
952
954
956
958
960
962
964
966
968
970
972
974
976
978
980
982
984
986
988
990
992
994
996
998
1000
1002
1004
1006
1008
1010
1012
1014
1016
1018
1020
1022
1024
1026
1028
1030
1032
1034
1036
1038
1040
1042
1044
1046
1048
1050
1052
1054
1056
1058
1060
1062
1064
1066
1068
1070
1072
1074
1076
1078
1080
1082
1084
1086
1088
1090
1092
1094
1096
1098
1100
1102
1104
1106
1108
1110
1112
1114
1116
1118
1120
1122
1124
1126
1128
1130
1132
1134
1136
1138
1140
1142
1144
1146
1148
1150
1152
1154
1156
1158
1160
1162
1164
1166
1168
1170
1172
1174
1176
1178
1180
1182
1184
1186
1188
1190
1192
1194
1196
1198
1200
1202
1204
1206
1208
1210
1212
1214
1216
1218
1220
1222
1224
1226
1228
1230
1232
1234
1236
1238
1240
1242
1244
1246
1248
1250
1252
1254
1256
1258
1260
1262
1264
1266
1268
1270
1272
1274
1276
1278
1280
1282
1284
1286
1288
1290
1292
1294
1296
1298
1300
1302
1304
1306
1308
1310
1312
1314
1316
1318
1320
1322
1324
1326
1328
1330
1332
1334
1336
1338
1340
1342
1344
1346
1348
1350
1352
1354
1356
1358
1360
1362
1364
1366
1368
1370
1372
1374
1376
1378
1380
1382
1384
1386
1388
1390
1392
1394
1396
1398
1400
1402
1404
1406
1408
1410
1412
1414
1416
1418
1420
1422
1424
1426
1428
1430
1432
1434
1436
1438
1440
1442
1444
1446
1448
1450
1452
1454
1456
1458
1460
1462
1464
1466
1468
1470
1472
1474
1476
1478
1480
1482
1484
1486
1488
1490
1492
1494
1496
1498
1500
1502
1504
1506
1508
1510
1512
1514
1516
1518
1520
1522
1524
1526
1528
1530
1532
1534
1536
1538
1540
1542
1544
1546
1548
1550
1552
1554
1556
1558
1560
1562
1564
1566
1568
1570
1572
1574
1576
1578
1580
1582
1584
1586
1588
1590
1592
1594
1596
1598
1600
1602
1604
1606
1608
1610
1612
1614
1616
1618
1620
1622
1624
1626
1628
1630
1632
1634
1636
1638
1640
1642
1644
1646
1648
1650
1652
1654
1656
1658
1660
1662
1664
1666
1668
1670
1672
1674
1676
1678
1680
1682
1684
1686
1688
1690
1692
1694
1696
1698
1700
1702
1704
1706
1708
1710
1712
1714
1716
1718
1720
1722
1724
1726
1728
1730
1732
1734
1736
1738
1740
1742
1744
1746
1748
1750
1752
1754
1756
1758
1760
1762
1764
1766
1768
1770
1772
1774
1776
1778
1780
1782
1784
1786
1788
1790
1792
1794
1796
1798
1800
1802
1804
1806
1808
1810
1812
1814
1816
1818
1820
1822
1824
1826
1828
1830
1832
1834
1836
1838
1840
1842
1844
1846
1848
1850
1852
1854
1856
1858
1860
1862
1864
1866
1868
1870
1872
1874
1876
1878
1880
1882
1884
1886
1888
1890
1892
1894
1896
1898
1900
1902
1904
1906
1908
1910
1912
1914
1916
1918
1920
1922
1924
1926
1928
1930
1932
1934
1936
1938
1940
1942
1944
1946
1948
1950
1952
1954
1956
1958
1960
1962
1964
1966
1968
1970
1972
1974
1976
1978
1980
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016
2018
2020
2022
2024
2026
2028
2030
2032
2034
2036
2038
2040
2042
2044
2046
2048
2050
2052
2054
2056
2058
2060
2062
2064
2066
2068
2070
2072
2074
2076
2078
2080
2082
2084
2086
2088
2090
2092
2094
2096
2098
2100
2102
2104
2106
2108
2110
2112
2114
2116
2118
2120
2122
2124
2126
2128
2130
2132
2134
2136
2138
2140
2142
2144
2146
2148
2150
2152
2154
2156
2158
2160
2162
2164
2166
2168
2170
2172
2174
2176
2178
2180
2182
2184
2186
2188
2190
2192
2194
2196
2198
2200
2202
2204
2206
2208
2210
2212
2214
2216
2218
2220
2222
2224
2226
2228
2230
2232
2234
2236
2238
2240
2242
2244
2246
2248
2250
2252
2254
2256
2258
2260
2262
2264
2266
2268
2270
2272
2274
2276
2278
2280
2282
2284
2286
2288
2290
2292
2294
2296
2298
2300
2302
2304
2306
2308
2310
2312
2314
2316
2318
2320
2322
2324
2326
2328
2330
2332
2334
2336
2338
2340
2342
2344
2346
2348
2350
2352
2354
2356
2358
2360
2362
2364
2366
2368
2370
2372
2374
2376
2378
2380
2382
2384
2386
2388
2390
2392
2394
2396
2398
2400
2402
2404
2406
2408
2410
2412
2414
2416
2418
2420
2422
2424
2426
2428
2430
2432
2434
2436
2438
2440
2442
2444
2446
2448
2450
2452
2454
2456
2458
2460
2462
2464
2466
2468
2470
2472
2474
2476
2478
2480
2482
2484
2486
2488
2490
2492
2494
2496
2498
2500
2502
2504
2506
2508
2510
2512
2514
2516
2518
2520
2522
2524
2526
2528
2530
2532
2534
2536
2538
2540
2542
2544
2546
2548
2550
2552
2554
2556
2558
2560
2562
2564
2566
2568
2570
2572
2574
2576
2578
2580
2582
2584
2586
2588
2590
2592
2594
2596
2598
2600
2602
2604
2606
2608
2610
2612
2614
2616
2618
2620
2622
2624
2626
2628
2630
2632
2634
2636
2638
2640
2642
2644
2646
2648
2650
2652
2654
2656
2658
2660
2662
2664
2666
2668
2670
2672
2674
2676
2678
2680
2682
2684
2686
2688
2690
2692
2694
2696
2698
2700
2702
2704
2706
2708
2710
2712
2714
2716
2718
2720
2722
2724
2726
2728
2730
2732
2734
2736
2738
2740
2742
2744
2746
2748
2750
2752
2754
2756
2758
2760
2762
2764
2766
2768
2770
2772
2774
2776
2778
2780
2782
2784
2786
2788
2790
2792
2794
2796
2798
2800
2802
2804
2806
2808
2810
2812
2814
2816
2818
2820
2822
2824
2826
2828
2830
2832
2834
2836
2838
2840
2842
2844
2846
2848
2850
2852
2854
2856
2858
2860
2862
2864
2866
2868
2870
2872
2874
2876
2878
2880
2882
2884
2886
2888
2890
2892
2894
2896
2898
2900
2902
2904
2906
2908
2910
2912
2914
2916
2918
2920
2922
2924
2926
2928
2930
2932
2934
2936
2938
2940
2942
2944
2946
2948
2950
2952
2954
2956
2958
2960
2962
2964
2966
2968
2970
2972
2974
2976
2978
2980
2982
2984
2986
2988
2990
2992
2994
2996
2998
3000
3002
3004
3006
3008
3010
3012
3014
3016
3018
3020
3022
3024
3026
3028
3030
3032
3034
3036
3038
3040
3042
3044
3046
3048
3050
3052
3054
3056
3058
3060
3062
3064
3066
3068
3070
3072
3074
3076
3078
3080
3082
3084
3086
3088
3090
3092
3094
3096
3098
3100
3102
3104
3106
3108
3110
3112
3114
3116
3118
3120
3122
3124
3126
3128
3130
3132
3134
3136
3138
3140
3142
3144
3146
3148
3150
3152
3154
3156
3158
3160
3162
3164
3166
3168
3170
3172
3174
3176
3178
3180
3182
3184
3186
3188
3190
3192
3194
3196
3198
3200
3202
3204
3206
3208
3210
3212
3214
3216
3218
3220
3222
3224
3226
3228
3230
3232
3234
3236
3238
3240
3242
3244
3246
3248
3250
3252
3254
3256
3258
3260
3262
3264
3266
3268
3270
3272
3274
3276
3278
3280
3282
3284
3286
3288
3290
3292
3294
3296
3298
3300
3302
3304
3306
3308
3310
3312
3314
3316
3318
3320
3322
3324
3326
3328
3330
3332
3334
3336
3338
3340
3342
3344
3346
3348
3350
3352
3354
3356
3358
3360
3362
3364
3366
3368
3370
3372
3374
3376
3378
3380
3382
3384
3386
3388
3390
3392
3394
3396
3398
3400
3402
3404
3406
3408
3410
3412
3414
3416
3418
3420
3422
3424
3426
3428
3430
3432
3434
3436
3438
3440
3442
3444
3446
3448
3450
3452
3454
3456
3458
3460
3462
3464
3466
3468
3470
3472
3474
3476
3478
3480
3482
3484
3486
3488
3490
3492
3494
3496
3498
3500
3502
3504
3506
3508
3510
3512
3514
3516
3518
3520
3522
3524
3526
3528
3530
3532
3534
3536
3538
3540
3542
3544
3546
3548
3550
3552
3554
3556
3558
3560
3562
3564
3566
3568
3570
3572
3574
3576
3578
3580
3582
3584
3586
3588
3590
3592
3594
3596
3598
3600
3602
3604
3606
3608
3610
3612
3614
3616
3618
3620
3622
3624
3626
3628
3630
3632
3634
3636
3638
3640
3642
3644
3646
3648
3650
3652
3654
3656
3658
3660
3662
3664
3666
3668
3670
3672
3674
3676
3678
3680
3682
3684
3686
3688
3690
3692
3694
3696
3698
3700
3702
3704
3706
3708
3710
3712
3714
3716
3718
3720
3722
3724
3726
3728
3730
3732
3734
3736
3738
3740
3742
3744
3746
3748
3750
3752
3754
3756
3758
3760
3762
3764
3766
3768
3770
3772
3774
3776
3778
3780
3782
3784
3786
3788
3790
3792
3794
3796
3798
3800
3802
3804
3806
3808
3810
3812
3814
3816
3818
3820
3822
3824
3826
3828
3830
3832
3834
3836
3838
3840
3842
3844
3846
3848
3850
3852
3854
3856
3858
3860
3862
3864
3866
3868
3870
3872
3874
3876
3878
3880
3882
3884
3886
3888
3890
3892
3894
3896
3898
3900
3902
3904
3906
3908
3910
3912
3914
3916
3918
3920
3922
3924
3926
3928
3930
3932
3934
3936
3938
3940
3942
3944
3946
3948
3950
3952
3954
3956
3958
3960
3962
3964
3966
3968
3970
3972
3974
3976
3978
3980
3982
3984
3986
3988
3990
3992
3994
3996
3998
4000
4002
4004
4006
4008
4010
4012
4014
4016
4018
4020
4022
4024
4026
4028
4030
4032
4034
4036
4038
4040
4042
4044
4046
4048
4050
4052
4054
4056
4058
4060
4062
4064
4066
4068
4070
4072
4074
4076
4078
4080
4082
4084
4086
4088
4090
4092
4094
4096
4098
4100
4102
4104
4106
4108
4110
4112
4114
4116
4118
4120
4122
4124
4126
4128
4130
4132
4134
4136
4138
4140
4142
4144
4146
4148
4150
4152
4154
4156
4158
4160
4162
4164
4166
4168
4170
4172
4174<br

and 506D and corresponding secure devices 414A and 414D by application programs 500 and operating system 502.

As indicated in Fig. 5, security kernel 504 is coupled to CPU SCU 416 and host bridge SCU 418 (e.g., via one or more device drivers). As will be described in detail below, CPU SCU 416 and host bridge SCU 418 control accesses to memory 406. CPU SCU 416 monitors all software-initiated accesses to memory 406, and host bridge SCU 418 monitors all hardware-initiated accesses to memory 406. Once configured by security kernel 504, CPU SCU 416 and host bridge SCU 418 allow only authorized accesses to memory 406.

40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
25510
25511
25512
25513
25514
25515
25516
25517
25518
25519
25520
25521
25522
25523
25524
25525
25526
25527
25528
25529
25530
25531
25532
25533
25534
25535
25536
25537
25538
25539
25540
25541
25542
25543
25544
25545
25546
25547
25548
25549
25550
25551
25552
25553
25554
25555
25556
25557
25558
25559
25560
25561
25562
25563
25564
25565
25566
25567
25568
25569
25570
25571
25572
25573
25574
25575
25576
25577
25578
25579
25580
25581
25582
25583
25584
25585
25586
25587
25588
25589
25590
25591
25592
25593
25594
25595
25596
25597
25598
25599
255100
255101
255102
255103
255104
255105
255106
255107
255108
255109
255110
255111
255112
255113
255114
255115
255116
255117
255118
255119
2551100
2551101
2551102
2551103
2551104
2551105
2551106
2551107
2551108
2551109
2551110
2551111
2551112
2551113
2551114
2551115
2551116
2551117
2551118
2551119
25511100
25511101
25511102
25511103
25511104
25511105
25511106
25511107
25511108
25511109
25511110
25511111
25511112
25511113
25511114
25511115
25511116
25511117
25511118
25511119
255111100
255111101
255111102
255111103
255111104
255111105
255111106
255111107
255111108
255111109
255111110
255111111
255111112
255111113
255111114
255111115
255111116
255111117
255111118
255111119
2551111100
2551111101
2551111102
2551111103
2551111104
2551111105
2551111106
2551111107
2551111108
2551111109
2551111110
2551111111
2551111112
2551111113
2551111114
2551111115
2551111116
2551111117
2551111118
2551111119
25511111100
25511111101
25511111102
25511111103
25511111104
25511111105
25511111106
25511111107
25511111108
25511111109
25511111110
25511111111
25511111112
25511111113
25511111114
25511111115
25511111116
25511111117
25511111118
25511111119
255111111100
255111111101
255111111102
255111111103
255111111104
255111111105
255111111106
255111111107
255111111108
255111111109
255111111110
255111111111
255111111112
255111111113
255111111114
255111111115
255111111116
255111111117
255111111118
255111111119
2551111111100
2551111111101
2551111111102
2551111111103
2551111111104
2551111111105
2551111111106
2551111111107
2551111111108
2551111111109
2551111111110
2551111111111
2551111111112
2551111111113
2551111111114
2551111111115
2551111111116
2551111111117
2551111111118
2551111111119
25511111111100
25511111111101
25511111111102
25511111111103
25511111111104
25511111111105
25511111111106
25511111111107
25511111111108
25511111111109
25511111111110
25511111111111
25511111111112
25511111111113
25511111111114
25511111111115
25511111111116
25511111111117
25511111111118
25511111111119
255111111111100
255111111111101
255111111111102
255111111111103
255111111111104
255111111111105
255111111111106
255111111111107
255111111111108
255111111111109
255111111111110
255111111111111
255111111111112
255111111111113
255111111111114
255111111111115
255111111111116
255111111111117
255111111111118
255111111111119
2551111111111100
2551111111111101
2551111111111102
2551111111111103
2551111111111104
2551111111111105
2551111111111106
2551111111111107
2551111111111108
2551111111111109
2551111111111110
2551111111111111
2551111111111112
2551111111111113
2551111111111114
2551111111111115
2551111111111116
2551111111111117
2551111111111118
2551111111111119
25511111111111100
25511111111111101
25511111111111102
25511111111111103
25511111111111104
25511111111111105
25511111111111106
25511111111111107
25511111111111108
25511111111111109
25511111111111110
25511111111111111
25511111111111112
25511111111111113
25511111111111114
25511111111111115
25511111111111116
25511111111111117
25511111111111118
25511111111111119
255111111111111100
255111111111111101
255111111111111102
255111111111111103
255111111111111104
255111111111111105
255111111111111106
255111111111111107
255111111111111108
255111111111111109
255111111111111110
255111111111111111
255111111111111112
255111111111111113
255111111111111114
255111111111111115
255111111111111116
255111111111111117
255111111111111118
255111111111111119
2551111111111111100
2551111111111111101
2551111111111111102
2551111111111111103
2551111111111111104
2551111111111111105
2551111111111111106
2551111111111111107
2551111111111111108
2551111111111111109
2551111111111111110
2551111111111111111
2551111111111111112
2551111111111111113
2551111111111111114
2551111111111111115
2551111111111111116
2551111111111111117
2551111111111111118
2551111111111111119
25511111111111111100
25511111111111111101
25511111111111111102
25511111111111111103
25511111111111111104
25511111111111111105
25511111111111111106
25511111111111111107
25511111111111111108
25511111111111111109
25511111111111111110
25511111111111111111
25511111111111111112
25511111111111111113
25511111111111111114
25511111111111111115
25511111111111111116
25511111111111111117
25511111111111111118
25511111111111111119
255111111111111111100
255111111111111111101
255111111111111111102
255111111111111111103
255111111111111111104
255111111111111111105
255111111111111111106
255111111111111111107
255111111111111111108
255111111111111111109
255111111111111111110
255111111111111111111
255111111111111111112
255111111111111111113
255111111111111111114
255111111111111111115
255111111111111111116
255111111111111111117
255111111111111111118
255111111111111111119
2551111111111111111100
2551111111111111111101
2551111111111111111102
2551111111111111111103
2551111111111111111104
2551111111111111111105
2551111111111111111106
2551111111111111111107
2551111111111111111108
2551111111111111111109
2551111111111111111110
2551111111111111111111
2551111111111111111112
2551111111111111111113
2551111111111111111114
2551111111111111111115
2551111111111111111116
2551111111111111111117
2551111111111111111118
2551111111111111111119
25511111111111111111100
25511111111111111111101
25511111111111111111102
25511111111111111111103
25511111111111111111104
25511111111111111111105
25511111111111111111106
25511111111111111111107
25511111111111111111108
25511111111111111111109
25511111111111111111110
25511111111111111111111
25511111111111111111112
25511111111111111111113
25511111111111111111114
25511111111111111111115
25511111111111111111116
25511111111111111111117
25511111111111111111118
25511111111111111111119
255111111111111111111100
255111111111111111111101
255111111111111111111102
255111111111111111111103
255111111111111111111104
255111111111111111111105
255111111111111111111106
255111111111111111111107
255111111111111111111108
255111111111111111111109
255111111111111111111110
255111111111111111111111
255111111111111111111112
255111111111111111111113
255111111111111111111114
255111111111111111111115
255111111111111111111116
255111111111111111111117
255111111111111111111118
255111111111111111111119
2551111111111111111111100
2551111111111111111111101
2551111111111111111111102
2551111111111111111111103
2551111111111111111111104
2551111111111111111111105
2551111111111111111111106
2551111111111111111111107
2551111111111111111111108
2551111111111111111111109
2551111111111111111111110
2551111111111111111111111
2551111111111111111111112
2551111111111111111111113
2551111111111111111111114
2551111111111111111111115
2551111111111111111111116
2551111111111111111111117
2551111111111111111111118
2551111111111111111111119
25511111111111111111111100
25511111111111111111111101
25511111111111111111111102
25511111111111111111111103
25511111111111111111111104
25511111111111111111111105
25511111111111111111111106
25511111111111111111111107
25511111111111111111111108
25511111111111111111111109
25511111111111111111111110
25511111111111111111111111
25511111111111111111111112
25511111111111111111111113
25511111111111111111111114
25511111111111111111111115
25511111111111111111111116
25511111111111111111111117
25511111111111111111111118
25511111111111111111111119
255111111111111111111111100
255111111111111111111111101
255111111111111111111111102
255111111111111111111111103
255111111111111111111111104
255111111111111111111111105
255111111111111111111111106
255111111111111111111111107
255111111111111111111111108
255111111111111111111111109
255111111111111111111111110
255111111111111111111111111
255111111111111111111111112
255111111111111111111111113
255111111111111111111111114
255111111111111111111111115
255111111111111111111111116
255111111111111111111111117
255111111111111111111111118
255111111111111111111111119
2551111111111111111111111100
2551111111111111111111111101
2551111111111111111111111102
2551111111111111111111111103
2551111111111111111111111104
2551111111111111111111111105
2551111111111111111111111106
2551111111111111111111111107
2551111111111111111111111108
2551111111111111111111111109
2551111111111111111111111110
2551111111111111111111111111
2551111111111111111111111112
2551111111111111111111111113
2551111111111111111111111114
2551111111111111111111111115
2551111111111111111111111116
2551111111111111111111111117
2551111111111111111111111118
2551111111111111111111111119
25511111111111111111111111100
25511111111111111111111111101
25511111111111111111111111102
25511111111111111111111111103
25511111111111111111111111104
25511111111111111111111111105
25511111111111111111111111106
25511111111111111111111111107
25511111111111111111111111108
25511111111111111111111111109
25511111111111111111111111110
25511111111

400 of Fig. 4, and operations of CPU SCU 416 and host bridge SCU 418 are governed by the contents of the set of SEM registers 610. SEM registers 610 are accessed (i.e., written to and/or read from) by security kernel 504 (Fig. 5). Computer system 400 of Fig. 4 may, for example, operate in the SEM when: (i) CPU 402 is an x86 processor operating in the x86 protected mode, (ii) memory paging is enabled, and (iii) the contents of SEM registers 610 specify SEM operation.

10 In the embodiment of Fig. 6, the set of SEM registers 610 includes a secure execution mode (SEM) bit. Operating modes of the computer system 400 (Fig. 4) include a “normal execution mode” and a “secure execution mode” (SEM). The computer system 400 normally operates in the normal execution mode. The set of SEM registers 610 is used to implement the secure execution mode (SEM) within the computer system 400. The SEM registers 610 are accessed (i.e., written to and/or read from) by the security kernel 504 (Fig. 5). The computer system 400 may, for example, operate in the secure execution mode (SEM) when: (i) the CPU 402 (Fig. 4) is an x86 processor operating in the x86 protected mode, (ii) memory paging is enabled, and (iii) the secure execution mode (SEM) bit is set to ‘1’.

15

20 In general, the contents of the set of control registers 608 govern operation of CPU 402. Accordingly, the contents of the set of control registers 608 govern operation of execution unit 600, MMU 602, cache unit 604, and/or BIU 606. The set of control registers 608 may include, for example, the multiple control registers of the x86 processor architecture.

Execution unit 600 of CPU 402 fetches instructions (e.g., x86 instructions) and data, executes the fetched instructions, and generates signals (e.g., address, data, and control

signals) during instruction execution. Execution unit 600 is coupled to cache unit 604, and may receive instructions from memory 406 (Fig. 4) via cache unit 604 and BIU 606.

Memory 406 (Fig. 4) of computer system 400 includes multiple memory locations, 5 each having a unique physical address. When operating in protected mode with paging enabled, an address space of CPU 402 is divided into multiple blocks called page frames or “pages.” As described above, only data corresponding to a portion of the pages is stored within memory 406 at any given time. In the embodiment of Fig. 6, address signals generated by execution unit 600 during instruction execution represent segmented (i.e., 10 “logical”) addresses. As described below, MMU 602 translates the segmented addresses generated by execution unit 600 to corresponding physical addresses of memory 406. MMU 602 provides the physical addresses to cache unit 604. Cache unit 604 is a relatively small storage unit used to store instructions and data recently fetched by execution unit 600. BIU 606 is coupled between cache unit 604 and host bridge 404, and is used to fetch instructions and data not present in cache unit 604 from memory 406 via host bridge 404. 15

Fig. 7 is a diagram of one embodiment of MMU 602 of Fig. 6. In the embodiment of Fig. 7, MMU 602 includes a segmentation unit 700, a paging unit 702, and selection logic 704 for selecting between outputs of segmentation unit 700 and paging unit 702 to produce a 20 physical address. As indicated in Fig. 7, segmentation unit 700 receives a segmented address from execution unit 600 and uses a well-known segmented-to-linear address translation mechanism of the x86 processor architecture to produce a corresponding linear address at an output. As indicated in Fig. 7, when enabled by a “PAGING” signal, paging unit 702 receives the linear addresses produced by segmentation unit 700 and produces corresponding 25 physical addresses at an output. The PAGING signal may mirror the paging flag (PG) bit in a

control register 0 (CR0) of the x86 processor architecture and of the set of control registers 608 (Fig. 6). When the PAGING signal is deasserted, memory paging is not enabled, and selection logic 704 produces the linear address received from segmentation unit 700 as the physical address.

5

When the PAGING signal is asserted, memory paging is enabled, and paging unit 702 translates the linear address received from segmentation unit 700 to a corresponding physical address using the above described linear-to-physical address translation mechanism 100 of the x86 processor architecture (Fig. 1). As described above, during the linear-to-physical address translation operation, the contents of the U/S bits of the selected page directory entry and the selected page table entry are logically ANDed to determine if the access to a page frame is authorized. Similarly, the contents of the R/W bits of the selected page directory entry and the selected page table entry are logically ANDed to determine if the access to the page frame is authorized. If the logical combinations of the U/S and R/W bits indicate the access to the page frame is authorized, paging unit 702 produces the physical address resulting from the linear-to-physical address translation operation. Selection logic 704 receives the physical address produced by paging unit 702, produces the physical address received from paging unit 702 as the physical address, and provides the physical address to cache unit 604.

20

On the other hand, if the logical combinations of the U/S and R/W bits indicate the access to the page frame 108 is not authorized, paging unit 702 does not produce a physical address during the linear-to-physical address translation operation. Instead, paging unit 702 asserts a page fault signal, and MMU 602 forwards the page fault signal to execution unit 600. In the x86 processor architecture, a page fault signal may, in some cases, indicate a protection violation. In response to the page fault signal, execution unit 600 may execute an

exception handler routine, and may ultimately halt the execution of one of the application programs 500 (Fig. 5) running when the page fault signal was asserted.

In the embodiment of Fig. 7, CPU SCU 416 is located within paging unit 702 of 5 MMU 602. Paging unit 702 may also include a translation lookaside buffer (TLB) for storing a relatively small number of recently determined linear-to-physical address translations.

Fig. 8 is a diagram of one embodiment of CPU SCU 416 of Fig. 7. In the embodiment of Fig. 8, CPU SCU 416 includes security check logic 800 coupled to the set of 10 SEM registers 610 (Fig. 6) and a security attribute table (SAT) entry buffer 802. As described below, SAT entries include additional security information above the U/S and R/W bits of page directory and page table entries corresponding to memory pages. Security check logic 800 uses the additional security information stored within a given SAT entry to prevent unauthorized software-initiated accesses to the corresponding memory page. SAT entry buffer 802 is used to store a relatively small number of SAT entries of recently accessed 15 memory pages.

As described above, the set of SEM registers 610 are used to implement a secure execution mode (SEM) within computer system 400 of Fig. 4. The contents of the set of 20 SEM registers 610 govern the operation of CPU SCU 416. Security check logic 800 receives information to be stored in SAT entry buffer 802 from MMU 602 via a communication bus indicated in Fig. 8. The security check logic 800 also receives a physical address produced by paging unit 702.

Figs. 9-11 will now be used to describe how additional security information of memory pages selected using address translation mechanism 100 of Fig. 1 is obtained within computer system 400 of Fig. 4. Fig. 9 is a diagram of one embodiment of a mechanism 900 for accessing a SAT entry of a selected memory page to obtain additional security information of the selected memory page. Mechanism 900 of Fig. 9 may be embodied within security check logic 800 of Fig. 8, and may be implemented when computer system 400 of Fig. 4 is operating in the SEM. Mechanism 900 involves a physical address 902 produced by paging mechanism 702 (Fig. 7) using address translation mechanism 100 of Fig. 1, a SAT directory 904, multiple SATs including a SAT 906, and a SAT base address register 908 of the set of SEM registers 610. SAT directory 104 and the multiple SATs, including SAT 906, are SEM data structures created and maintained by security kernel 504 (Fig. 5). As described below, SAT directory 104 (when present) and any needed SAT is copied into memory 406 before being accessed.

10
15
20

SAT base address register 908 includes a present (P) bit which indicates the presence of a valid SAT directory base address within SAT base address register 908. The highest ordered (i.e., most significant) bits of SAT base address register 908 are reserved for the SAT directory base address. The SAT directory base address is a base address of a memory page containing SAT directory 904. If P=1, the SAT directory base address is valid, and SAT tables specify the security attributes of memory pages. If P=0, the SAT directory base address is not valid, no SAT tables exist, and security attributes of memory pages are determined by a SAT default register.

25 Fig. 10 is a diagram of one embodiment of the SAT default register 1000. In the embodiment of Fig. 10, SAT default register 1000 includes a secure page (SP) bit. The SP bit

indicates whether or not all memory pages are secure pages. For example, if SP=0 all memory pages may not be secure pages, and if SP=1 all memory pages may be secure pages.

Referring back to Fig. 9 and assuming the P bit of SAT base address register 908 is a 5 '1', physical address 902 produced by paging logic 702 (Fig. 7) is divided into three portions to access the SAT entry of the selected memory page. As described above, the SAT directory base address of SAT base address register 908 is the base address of the memory page containing SAT directory 904. SAT directory 904 includes multiple SAT directory entries, including a SAT directory entry 910. Each SAT directory entry may have a corresponding SAT in memory 406. An "upper" portion of physical address 902, including the highest ordered or most significant bits of physical address 902, is used as an index into SAT directory 904. SAT directory entry 910 is selected from within SAT directory 904 using the SAT directory base address of SAT base address register 908 and the upper portion of physical address 902.

10
15
20
25

20

Fig. 11 is a diagram of one embodiment of a SAT directory entry format 1100. In accordance with Fig. 11, each SAT directory entry includes a present (P) bit which indicates the presence of a valid SAT base address within the SAT directory entry. In the embodiment of Fig. 11, the highest ordered (i.e., the most significant) bits of each SAT directory entry are reserved for a SAT base address. The SAT base address is a base address of a memory page containing a corresponding SAT. If P=1, the SAT base address is valid, and the corresponding SAT is stored in memory 406.

If P=0, the SAT base address is not valid, and the corresponding SAT does not exist 25 in memory 406 and must be copied into memory 406 from a storage device (e.g., a disk

drive). If $P=0$, security check logic 800 may signal a page fault to logic within paging unit 702, and MMU 602 may forward the page fault signal to execution unit 600 (Fig. 6). In response to the page fault signal, execution unit 600 may execute a page fault handler routine which retrieves the needed SAT from the storage device and stores the needed SAT in 5 memory 406. After the needed SAT is stored in memory 406, the P bit of the corresponding SAT directory entry is set to '1', and mechanism 900 is continued.

Referring back to Fig. 9, a "middle" portion of physical address 902 is used as an index into SAT 906. SAT entry 906 is thus selected within SAT 906 using the SAT base address of SAT directory entry 910 and the middle portion of physical address 902. Fig. 12 is a diagram of one embodiment of a SAT entry format 1200. In the embodiment of Fig. 12, each SAT entry includes a secure page (SP) bit. The SP bit indicates whether or not the selected memory page is a secure page. For example, if $SP=0$ the selected memory page may not be a secure page, and if $SP=1$ the selected memory page may be a secure page.

40
41
42
43
44
45

20

BIU 606 (Fig. 6) retrieves needed SEM data structure entries from memory 406, and provides the SEM data structure entries to MMU 602. Referring back to Fig. 8, security check logic 800 receives SEM data structure entries from MMU 602 and paging unit 702 via the communication bus. As described above, SAT entry buffer 802 is used to store a relatively small number of SAT entries of recently accessed memory pages. Security check logic 800 stores a given SAT entry in SAT entry buffer 802, along with a "tag" portion of the corresponding physical address.

25

During a subsequent memory page access, security check logic 800 may compare a

"tag" portion of a physical address produced by paging unit 702 to tag portions of physical

addresses corresponding to SAT entries stored in SAT entry buffer 802. If the tag portion of the physical address matches a tag portion of a physical address corresponding to a SAT entry stored in SAT entry buffer 802, security check logic 800 may access the SAT entry in SAT entry buffer 802, eliminating the need to perform the process of Fig. 9 to obtain the SAT entry from memory 406. Security kernel 504 (Fig. 5) modifies the contents of SAT base address register 908 in CPU 402 (e.g., during context switches). In response to modifications of SAT base address register 908, security check logic 800 of CPU SCU 416 may flush SAT entry buffer 802.

40
45
50
55

20

When computer system 400 of Fig. 4 is operating in the SEM, security check logic 800 receives the current privilege level (CPL) of the currently executing task (i.e., the currently executing instruction). Alternately, security check logic 800 may receive the value of the secure execution mode (SEM) bit stored in the set of SEM registers 610 (Fig. 6). The CPL of the currently executing task, or the value of the secure execution mode (SEM) bit, represents a security attribute of the currently executing instruction. Security check logic 800 also receives the page directory entry (PDE) U/S bit, the PDE R/W bit, the page table entry (PTE) U/S bit, and the PTE R/W bit of a selected memory page within which a physical address resides. Security check logic 800 uses the above information, along with the SP bit of the SAT entry corresponding to the selected memory page, to determine if memory 406 access is authorized.

25

CPU 402 of Fig. 6 may be an x86 processor, and may include a code segment (CS) register, one of the 16-bit segment registers of the x86 processor architecture. Each segment register selects a 64k block of memory, called a segment. In the protected mode with paging enabled, the CS register is loaded with a segment selector that indicates an executable

segment of memory 406. The highest ordered (i.e., most significant) bits of the segment selector are used to store information indicating a segment of memory including a next instruction to be executed by execution unit 600 of CPU 402 (Fig. 6). An instruction pointer (IP) register is used to store an offset into the segment indicated by the CS register. The 5 CS:IP pair indicate a segmented address of the next instruction. The two lowest ordered (i.e., least significant) bits of the CS register are used to store a value indicating a current privilege level (CPL) of a task currently being executed by execution unit 600 (i.e., the CPL of the current task).

40 Table 1 below illustrates exemplary rules for CPU-initiated (i.e., software-initiated) memory accesses when computer system 400 of Fig. 4 is operating in the SEM, and security check logic 800 receives the current privilege level (CPL) of the currently executing task. CPU SCU 416 (Figs. 4-8) and security kernel 504 (Fig. 5) would expectedly work together to implement the rules of Table 1 when computer system 400 of Fig. 4 is operating in the SEM to provide additional security for data stored in memory 406 above data security provided by 45 operating system 502 (Fig. 5).

Table 1. Exemplary Rules For Software-Initiated Memory Accesses

When Computer System 400 Of Fig. 4 Is Operating In The SEM.

20

Currently Executing Instruction	Selected Memory Page	Permitted			
<u>SP</u>	<u>CPL</u>	<u>SP</u>	<u>U/S R/W</u>	<u>Access</u>	<u>Remarks</u>

25

1	0	X X 1(R/W)	R/W	Full access granted. (Typical accessed page contents: security kernel and SEM data structures.)
5	1 0	X X 0(R)	Read Only	Write attempt causes page fault; if selected memory page is a secure page (SP=1), a SEM Security Exception is signaled instead of page fault.
10	1 3	1 1 (U) 1	R/W	Standard protection mechanisms apply. (Typical accessed page contents: high security applets.)
15	1 3	1 0 (S) X	None	Access causes page fault. (Typical accessed page contents: security kernel and SEM data structures.)
20	1 3	0 0 1	None	Access causes page fault. (Typical accessed page contents: OS kernel and Ring 0 device drivers.)
25	0 0	1 X X	None	Access causes SEM security exception.

0 0	0 1 1	R/W	Standard protection mechanisms apply. (Typical accessed page contents: high security applets.)
0 3	X 0 X	None	Access causes page fault; if selected memory page is a secure page (SP=1), a SEM Security Exception is raised instead of page fault.
0 3	0 1 1	R/W	Standard protection mechanisms apply. (Typical accessed page contents: applications.)

In Table 1 above, the SP bit of the currently executing instruction is the SP bit of the

SAT entry corresponding to the memory page containing the currently executing instruction.

The U/S bit of the selected memory page is the logical AND of the PDE U/S bit and the PTE U/S bit of the selected memory page. The R/W bit of the selected memory page is the logical

20 AND of the PDE R/W bit and the PTE R/W bit of the selected memory page. The symbol “X” signifies a “don’t care”: the logical value may be either a ‘0’ or a ‘1’.

It is noted that, as shown in Fig. 8 and described above, security check logic 800 may

receive the value of the secure execution mode (SEM) bit, stored in the set of SEM registers

25 610 of Fig. 6, in place of the CPL of the currently executing task. In this situation, the

security check logic 800 may generate the page fault signal dependent upon the value of the SEM bit.

Referring back to Fig. 8, security check logic 800 of CPU SCU 416 produces a page fault signal and a “SEM SECURITY EXCEPTION” signal, and provides the page fault and the SEM SECURITY EXCEPTION signals to logic within paging unit 702. When security check logic 800 asserts the page fault signal, MMU 602 forwards the page fault signal to execution unit 600 (Fig. 6). In response to the page fault signal, execution unit 600 may use the well-known interrupt descriptor table (IDT) vectoring mechanism of the x86 processor architecture to access and execute a page fault handler routine.

10
15

20

25

When security check logic 800 asserts the SEM SECURITY EXCEPTION signal, MMU 602 forwards the SEM SECURITY EXCEPTION signal to execution unit 600. Unlike normal processor exceptions which use the use the IDT vectoring mechanism of the x86 processor architecture, a different vectoring method may be used to handle SEM security exceptions. SEM security exceptions may be dispatched through a pair of registers (e.g., model specific registers or MSRs) similar to the way x86 “SYSENTER” and “SYSEXIT” instructions operate. The pair of registers may be “security exception entry point” registers, and may define a branch target address for instruction execution when a SEM security exception occurs. The security exception entry point registers may define the code segment (CS), then instruction pointer (IP, or the 64-bit version RIP), stack segment (SS), and the stack pointer (SP, or the 64-bit version RSP) values to be used on entry to a SEM security exception handler. Under software control, execution unit 600 (Fig. 6) may push the previous SS, SP/RSP, EFLAGS, CS, and IP/RIP values onto a new stack to indicate where the exception occurred. In addition, execution unit 600 may push an error code onto the

stack. It is noted that a normal return from interrupt (IRET) instruction may not be used as the previous SS and SP/RSP values are always saved, and a stack switch is always accomplished, even if a change in CPL does not occur. Accordingly, a new instruction may be defined to accomplish a return from the SEM security exception handler.

5

Fig. 13 is a diagram of one embodiment of host bridge 404 of Fig. 4. In the embodiment of Fig. 13, host bridge 404 includes a host interface 1300, bridge logic 1302, host bridge SCU 418, a memory controller 1304, and a device bus interface 1306. Host interface 1300 is coupled to CPU 402, and device bus interface 1306 is coupled to device bus 408. Bridge logic 1302 is coupled between host interface 1300 and device bus interface 1306. Memory controller 1304 is coupled to memory 406, and performs all accesses to memory 406. Host bridge SCU 418 is coupled between bridge logic 1302 and memory controller 1304. As described above, host bridge SCU 418 controls access to memory 406 via device bus interface 1306. Host bridge SCU 418 monitors all accesses to memory 406 via device bus interface 1306, and allows only authorized accesses to memory 406.

40
41
42
43
44
45

Fig. 14 is a diagram of one embodiment of host bridge SCU 418 of Fig. 13. In the embodiment of Fig. 14, host bridge SCU 418 includes security check logic 1400 coupled to a set of SEM registers 1402 and a SAT entry buffer 1404. The set of SEM registers 1402 govern the operation of security check logic 1400, and includes a second SAT base address register 908 of Fig. 9. The second SAT base address register 908 of the set of SEM registers 1402 may be an addressable register. When security kernel 504 (Fig. 5) modifies the contents of SAT base address register 908 in the set of SEM registers 610 of CPU 402 (e.g., during a context switch), security kernel 504 may also write the same value to the second SAT base address register 908 in the set of SEM registers 1402 of host bridge SCU 418. In

response to modifications of the second SAT base address register 908, security check logic 1400 of host bridge SCU 418 may flush SAT entry buffer 1404.

Security check logic 1400 receives memory access signals of memory accesses initiated by hardware device units 414A-414D (Fig. 4) via device bus interface 1306 and bridge logic 1302 (Fig. 13). The memory access signals convey physical addresses from hardware device units 414A-414D, and associated control and/or data signals. Security check logic 1400 may embody mechanism 900 (Fig. 9) for obtaining SAT entries of corresponding memory pages, and may implement mechanism 900 when computer system 400 of Fig. 4 is operating in the SEM. SAT entry buffer 1404 is similar to SAT entry buffer 802 of CPU SCU 416 (Fig. 8) described above, and is used to store a relatively small number of SAT entries of recently accessed memory pages.

10
05
05

When computer system 400 of Fig. 4 is operating in SEM, security check logic 1400 of Fig. 14 uses additional security information of a SAT entry associated with a selected memory page to determine if a given hardware-initiated memory access is authorized. If the given hardware-initiated memory access is authorized, security check logic 1400 provides the memory access signals (i.e., address signals conveying a physical address and the associated control and/or data signals) of the memory access to memory controller 1304. Memory controller 1304 uses the physical address and the associated control and/or data signals to access memory 406. If memory 406 access is a write access, data conveyed by the data signals is written to memory 406. If memory 406 access is a read access, memory controller 1304 reads data from memory 406, and provides the resulting read data to security check logic 1400. Security check logic 1400 forwards the read data to bridge logic 1302, and bridge logic 1302 provides the data to device bus interface 1306.

If, on the other hand, the given hardware-initiated memory access is not authorized, security check logic 1400 does not provide the physical address and the associated control and/or data signals of memory 406 accesses to memory controller 1304. If the unauthorized 5 hardware-initiated memory access is a memory write access, security check logic 1400 may signal completion of the write access and discard the write data, leaving memory 406 unchanged. Security check logic 1400 may also create a log entry in a log (e.g., set or clear one or more bits of a status register) to document the security access violation. Security kernel 504 may periodically access the log to check for such log entries. If the unauthorized 10 hardware-initiated memory access is a memory read access, security check logic 1400 may return a false result (e.g., all “F”s) to device bus interface 1306 via bridge logic 1302 as the read data. Security check logic 1400 may also create a log entry as described above to document the security access violation.

100

105

20

25

Fig. 15 is a flow chart of one embodiment of a method 1500 for providing access security for a memory used to store data arranged within multiple memory pages. Method 1500 reflects the exemplary rules of Table 1 for CPU-initiated (i.e., software-initiated) memory accesses when computer system 400 of Fig. 4 is operating in the SEM. Method 1500 may be embodied within MMU 602 (Figs. 6-7). During a step 1502 of method 1500, a linear address produced during execution of an instruction is received, along with a security attribute of the instruction (e.g., a CPL of a task including the instruction, or a value of the SEM bit stored in the set of SEM registers 610 of Fig. 6). The instruction resides in a memory page. During a step 1504, the linear address is used to access at least one paged memory data structure located in the memory (e.g., a page directory and a page table) to obtain a base address of a selected memory page and security attributes of the selected 20 memory page. During a step 1506, the security attributes of the selected memory page are used to determine if the memory access is authorized. If the memory access is unauthorized, the memory access is denied and a log entry is created in a log (e.g., set or clear one or more bits of a status register) to document the security access violation. If the memory access is authorized, the memory access is granted and the base address of the selected memory page is provided to a memory controller 1304. During a step 1508, the memory controller 1304 provides the base address of the selected memory page and the security attributes of the selected memory page to a memory 406. The memory 406 provides the memory access to the memory controller 1304 based on the base address and the security attributes. During a step 1510, the memory controller 1304 provides the memory access to a device bus interface 1306 via bridge logic 1302. The device bus interface 1306 provides the memory access to a device 1308. The device 1308 performs the memory access based on the memory access provided by the device bus interface 1306. During a step 1512, the device 1308 provides the memory access to a host processor 1310. The host processor 1310 performs the memory access based on the memory access provided by the device 1308. During a step 1514, the host processor 1310 provides the memory access to a user application 1312. The user application 1312 performs the memory access based on the memory access provided by the host processor 1310. During a step 1516, the user application 1312 provides the memory access to a user 1314. The user 1314 performs the memory access based on the memory access provided by the user application 1312.

memory page. The security attributes of the selected memory page may include, for example, a U/S bit and a R/W bit of a page directory entry and a U/S bit and a R/W bit of a page table entry.

5 During a decision step 1506, the security attribute of the instruction and the security attributes of the selected memory page are used to determine whether or not the access is authorized. If the access is authorized, the base address of the selected memory page and an offset are combined during a step 1508 to produce a physical address within the selected memory page. If the access is not authorized, a fault signal (e.g., a page fault signal) is generated during a step 1510.

H0
H1
H2
H3
H4
H5

20

During a step 1512 following step 1508, at least one security attribute data structure located in the memory (e.g., SAT directory 904 of Fig. 9 and a SAT) is accessed using the physical address of the selected memory page to obtain an additional security attribute of the first memory page and an additional security attribute of the selected memory page. The additional security attribute of the first memory page may include, for example, a secure page (SP) bit as described above, wherein the SP bit indicates whether or not the first memory page is a secure page. Similarly, the additional security attribute of the selected memory page may include a secure page (SP) bit, wherein the SP bit indicates whether or not the selected memory page is a secure page.

The fault signal is generated during a step 1514 dependent upon the security attribute of the instruction, the additional security attribute of the first memory page, the security attributes of the selected memory page, and the additional security attribute of the selected

memory page. It is noted that steps 1512 and 1514 of method 1500 may be embodied within CPU SCU 416 (Figs. 4-8).

Table 2 below illustrates exemplary rules for memory page accesses initiated by device hardware units 414A-414D (i.e., hardware-initiated memory accesses) when computer system 400 of Fig. 4 is operating in the SEM. Such hardware-initiated memory accesses may be initiated by bus mastering circuitry within device hardware units 414A-414D, or by DMA devices at the request of device hardware units 414A-414D. Security check logic 1400 may implement the rules of Table 2 when computer system 400 of Fig. 4 is operating in the SEM to provide additional security for data stored in memory 406 above data security provided by operating system 502 (Fig. 5). In Table 2 below, the “target” memory page is the memory page within which a physical address conveyed by memory access signals of a memory access resides.

Table 2. Exemplary Rules For Hardware-Initiated Memory Accesses

When Computer system 400 Of Fig. 4 Is Operating In The SEM.

	Particular			
	Memory			
	Page			
		Access		
		<u>SP</u>	<u>Type</u>	<u>Action</u>
		0	R/W	The access completes as normal.
25	1	Read		The access is completed returning

all “F”s instead of actual memory contents. The unauthorized access may be logged.

5	1	Write	The access is completed but write data is discarded. Memory contents remain unchanged. The unauthorized access may be logged.
---	---	-------	---

10 In Table 2 above, the SP bit of the target memory page is obtained by host bridge SCU 418 using the physical address of the memory access and the above described mechanism 900 of Fig. 9 for obtaining SAT entries of corresponding memory pages.

15 As indicated in Fig. 2, when SP=1 indicating the target memory page is a secure page, the memory access is unauthorized. In this situation, security check logic 1400 (Fig. 14) does not provide the memory access signals to the memory controller. A portion of the memory access signals (e.g., the control signals) indicate a memory access type, and wherein the memory access type is either a read access or a write access. When SP=1 and the memory access signals indicate the memory access type is a read access, the memory access is an 20 unauthorized read access, and security check logic 1400 responds to the unauthorized read access by providing all “F”s instead of actual memory contents (i.e., bogus read data). Security check logic 1400 may also respond to the unauthorized read access by logging the unauthorized read access as described above.

When SP=1 and the memory access signals indicate the memory access type is a write access, the memory access is an unauthorized write access. In this situation, security check logic 1400 responds to the unauthorized write access by discarding write data conveyed by the memory access signals. Security check logic 1400 may also respond to the unauthorized write access by logging the unauthorized write access as described above.

10
15
20
25

20

Fig. 16 is a flow chart of one embodiment of a method 1600 for providing access security for a memory used to store data arranged within multiple memory pages. Method 1600 reflects the exemplary rules of Table 2 for hardware-initiated memory accesses when computer system 400 of Fig. 4 is operating in the SEM. Method 1600 may be embodied within host bridge 404 (Figs. 4 and 13-14). During a step 1602 of method 1600, memory access signals of a memory access are received, wherein the memory access signals convey a physical address within a target memory page. As described above, the memory access signals may be produced by a device hardware unit. The physical address is used to access at least one security attribute data structure located in the memory to obtain a security attribute of the target memory page during a step 1604. The at least one security attribute data structure may include, for example, a SAT directory (e.g., SAT directory 904 in Fig. 9) and at least one SAT (e.g., SAT 906 in Fig. 9), and the additional security attribute of the target memory page may include a secure page (SP) bit as described above which indicates whether or not the target memory page is a secure page. During a step 1606, the memory is accessed using the memory access signals dependent upon the security attribute of the target memory page.

25

The particular embodiments disclosed above are illustrative only, as the invention

may be modified and practiced in different but equivalent manners apparent to those skilled

in the art having the benefit of the teachings herein. Furthermore, no limitations are intended to the details of construction or design herein shown, other than as described in the claims below. It is therefore evident that the particular embodiments disclosed above may be altered or modified and all such variations are considered within the scope and spirit of the invention.

- 5 Accordingly, the protection sought herein is as set forth in the claims below.